Taxi Duration Prediction

Table of Contents

Objectives

The main objective of this report is to use the yellow taxicab dataset and predict trip duration in minutes. We will also explore the most important features involved in the predictions.

Brief description of the data set and a summary of its attributes

TLC Trip Record Data has 12 years (2009 –2020) worth of Data available but I have decided to analyze a subset of the 2015.

In 2015, passengers took nearly 300 million yellow cab rides in New York City. Working with the complete dataset for all these rides would require considerable time and computational resources. The 12 data files used represent two percent of the total trips sampled at random from each month.

I chose this data because it is useful for real-world applications such as:

- Predicting taxi duration for a trip
- Allocating taxi to zones/regions based on demand.

Below is the summary of the data attributes as described here

Attribute / Column	Description
VendorID	A code indicating the TPEP provider that provided the record. **1= Creative Mobile Technologies, LLC; 2= VeriFone Inc.**
tpep_pickup_datetime	The date and time when the meter was engaged.
tpep_dropoff_datetime	The date and time when the meter was disengaged.
Passenger_count	The number of passengers in the vehicle. This is a driver-entered value.
Trip_distance	The elapsed trip distance in miles reported by the taximeter.
PULocationID	TLC Taxi Zone in which the taximeter was engaged.
DOLocationID	TLC Taxi Zone in which the taximeter was disengaged.
RateCodeID	The final rate code in effect at the end of the trip. **1= Standard rate** **2=JFK** **3=Newark** **4=Nassau or Westchester** **5=Negotiated fare** **6=Group ride**
Store_and_fwd_flag	This flag indicates whether the trip record was held in vehicle memory before sending to the vendor, aka “store and forward,” because the vehicle did not have a connection to the server. **Y= store and forward trip** **N= not a store and forward trip**
Payment_type	A numeric code signifying how the passenger paid for the trip. **1= Credit card** **2= Cash** **3= No charge** **4= Dispute** **5= Unknown** **6= Voided trip**
Fare_amount	The time-and-distance fare calculated by the meter.
Extra	Miscellaneous extras and surcharges. Currently, this only includes the $0.50 and $1 rush hour and overnight charges.
MTA_tax	$0.50 MTA tax that is automatically triggered based on the metered rate in use.
Improvement_surcharge	$0.30 improvement surcharge assessed trips at the flag drop. The improvement surcharge began being levied in 2015.
Tip_amount	Tip amount –This field is automatically populated for credit card tips. Cash tips are not included.
Tolls_amount	Total amount of all tolls paid in trip.
Total_amount	The total amount charged to passengers. Does not include cash tips.

Data Exploration

Joining the 12 files provided resulted in 2, 922, 266 instances in the dataset with no missing values. We have 12 numerical features, 4 categorical features, 2 datetime features and 1 integer. For easy visualization, I converted *VendorID, RateCodeID and payment_type* to their corresponding values.

Summary of some the numerical attributes shows that this data is not perfect and therefore needs some cleaning for example there is no way we can have a negative tip or zero passengers.

	Passenger_count	Trip_distance	Fare_amount	Tip_amount
min	0.0	0.0	-150.00	-2.7
max	9.0	14680110.0	410266.86	650

Data Cleaning and Feature engineering

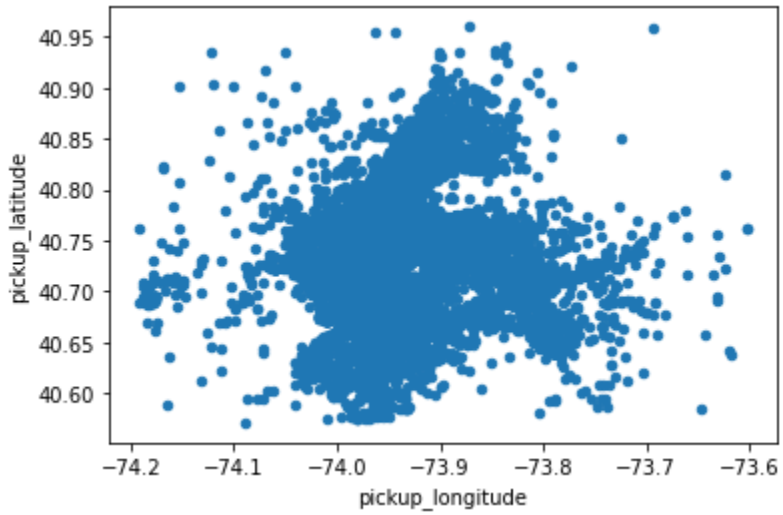
For data preprocessing/cleaning

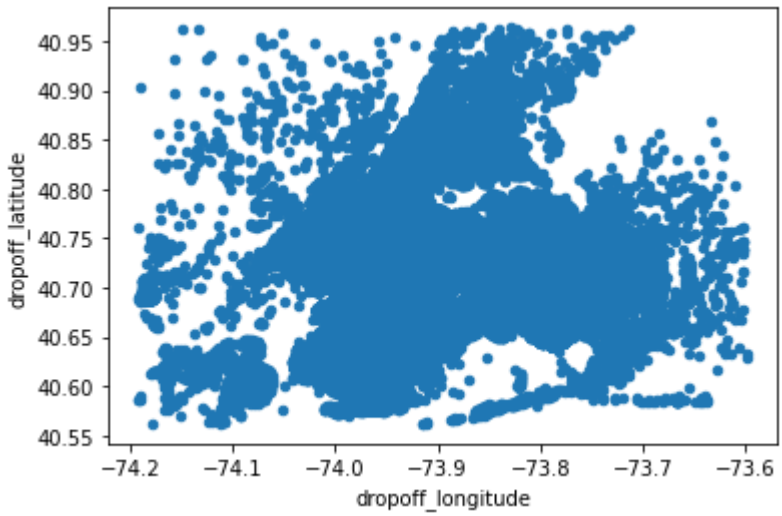
- Charges and trip information that are negative, as well as charges inconsistent with expected values are considered incorrect. In addition, trips with pickup or drop off locations outside a geographic region of interest are removed.
- Only keep trips with valid passenger and distance information.
- Remove trips missing valid pickup or drop off locations.
- Remove trips with invalid Rate code.

For Feature engineering, the following features were added.

- ***duration*** - Length of the trip, in minutes, calculated from the pickup and drop off times.
- ***avespeed*** - Average speed, in mph, calculated from the distance and duration values.
- ***time_of_day*** - This feature represents pick up time as the elapsed time since midnight in decimal hours (e.g. 7:10 am becomes 7.1667). The output is a duration vector with units of hours.
- ***day_of_week*** - This feature is a categorical array indicating the day of the week the trip began, in long format (e.g. 'Monday').

After removing invalid trip information, we can now visualize the features again.





Data Preparation for Machine Learning

Feature Selection

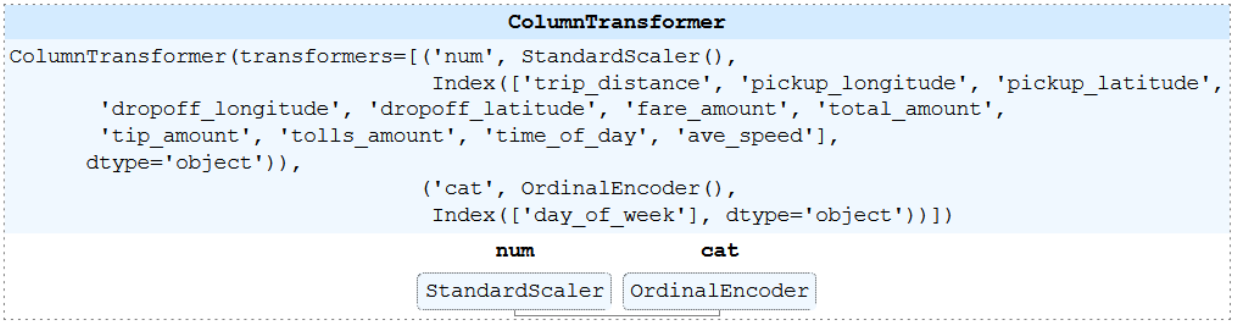
We chose the target variable to be *duration* as we want to predict trip duration. To avoid using all the 22 columns as features, we tried to find the correlation between each numerical feature and the target variable, and the result is as below:

Features	Correlation with duration
total_amount	**0.851412**
fare_amount	**0.878996**
trip_distance	**0.781429**
tip_amount	**0.491806**
tolls_amount	**0.449873**
pickup_longitude	**0.367773**
dropoff_longitude	**0.238447**
ave_speed	**0.165792**
time_of_day	**0.031575**
passenger_count	0.015956
improvement_surcharge	0.014914
extra	**-0.056300**
mta_tax	**-0.081551**
dropoff_latitude	**-0.183337**
pickup_latitude	**-0.232437**

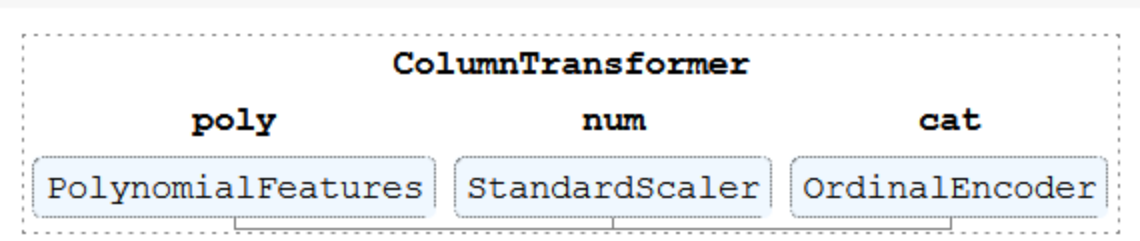
We opted to use the highlighted numerical features and ***day_of_week*** as the only categorical feature.

Feature transformation

We applied two transformations on the features: feature scaling for the numerical features and ordinal encoding for the categorical feature. We built below transformation pipeline using scikit-learn's ***ColumnTransformer*** class.



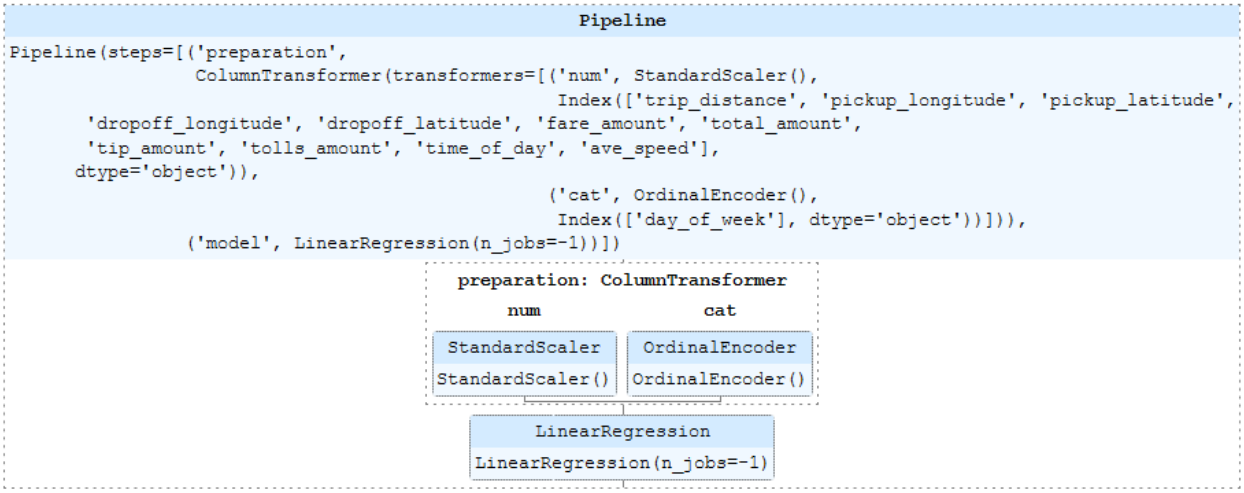
We also built a second pipeline to explore the effects of adding polynomial features on our model.



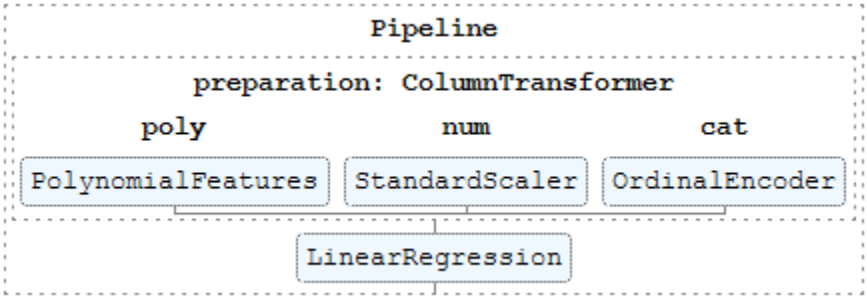
Model Training and Evaluation

We trained three different regression models:

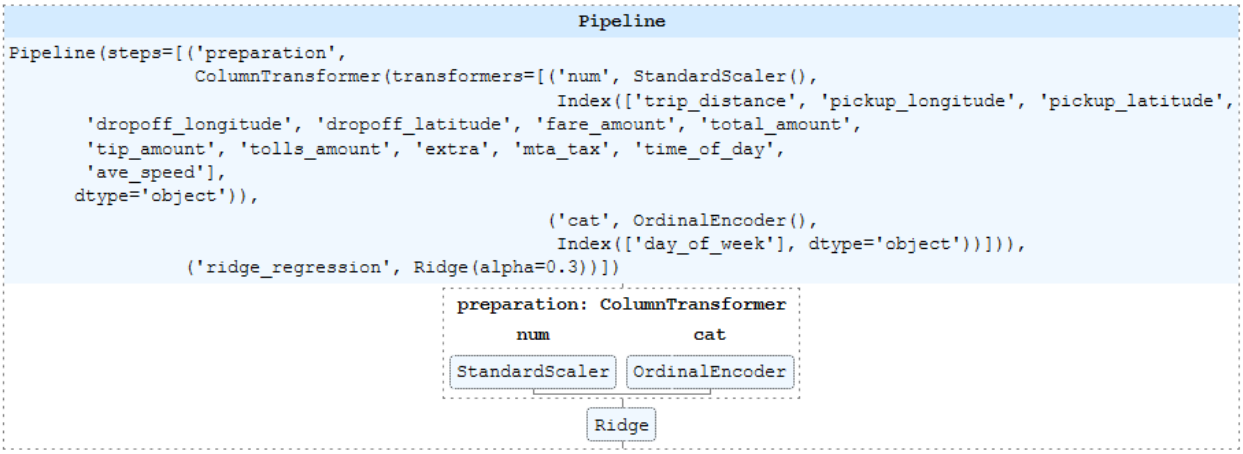
- Simple Linear Regression model without polynomial features



- Linear Regression model with polynomial features



- Ridge model without polynomial features and used ***GridSearchCV*** to select the best alpha to avoid overfitting.



We split our data into three: train (1,706,325 rows), validation (682,530 rows) and test (455,020 rows).

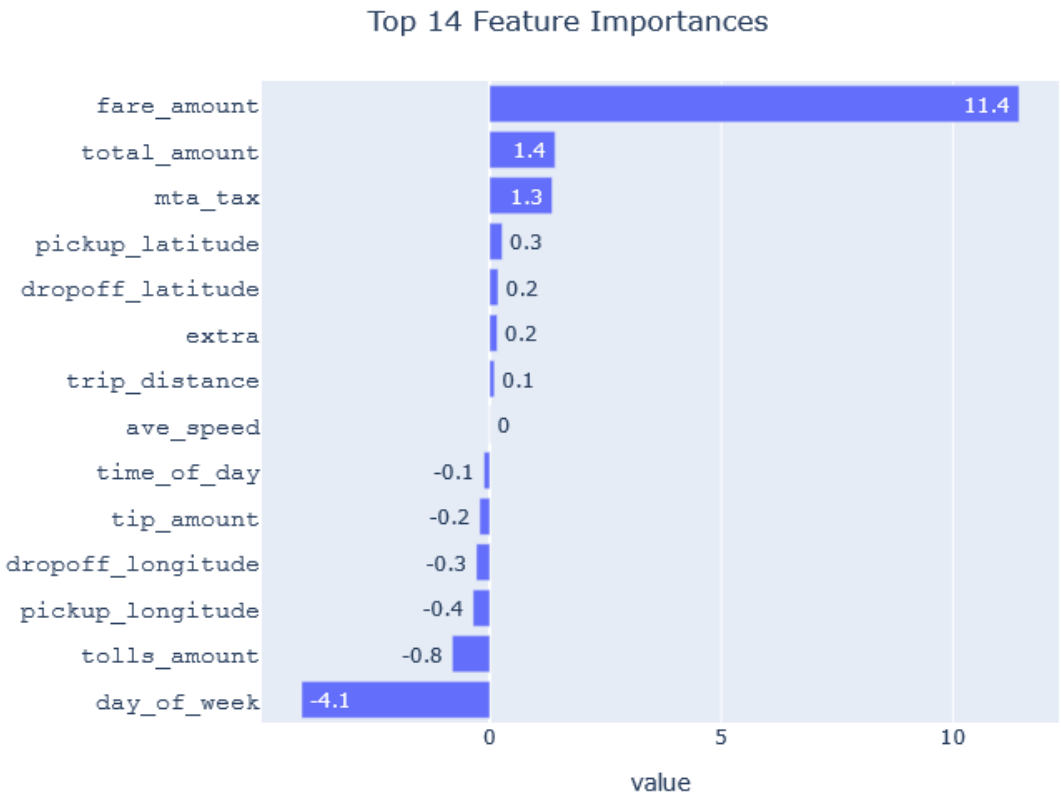
We evaluated the model using r2_score and rmse and obtain below outputs:

	Linear Regression		**Linear Regression w/Poly**		**Ridge Regression**	
	Score	Rmse	Score	Rmse	Score	Rmse
Train	0.893005	3.359505	**0.973563**	**1.734226**	0.893005	3.359505
Validation	0.893268	3.349676	**0.973258**	**1.740619**	0.893267	3.349676
Test	0.893786	3.338581	**0.973844**	**1.721172**	0.893786	3.338583

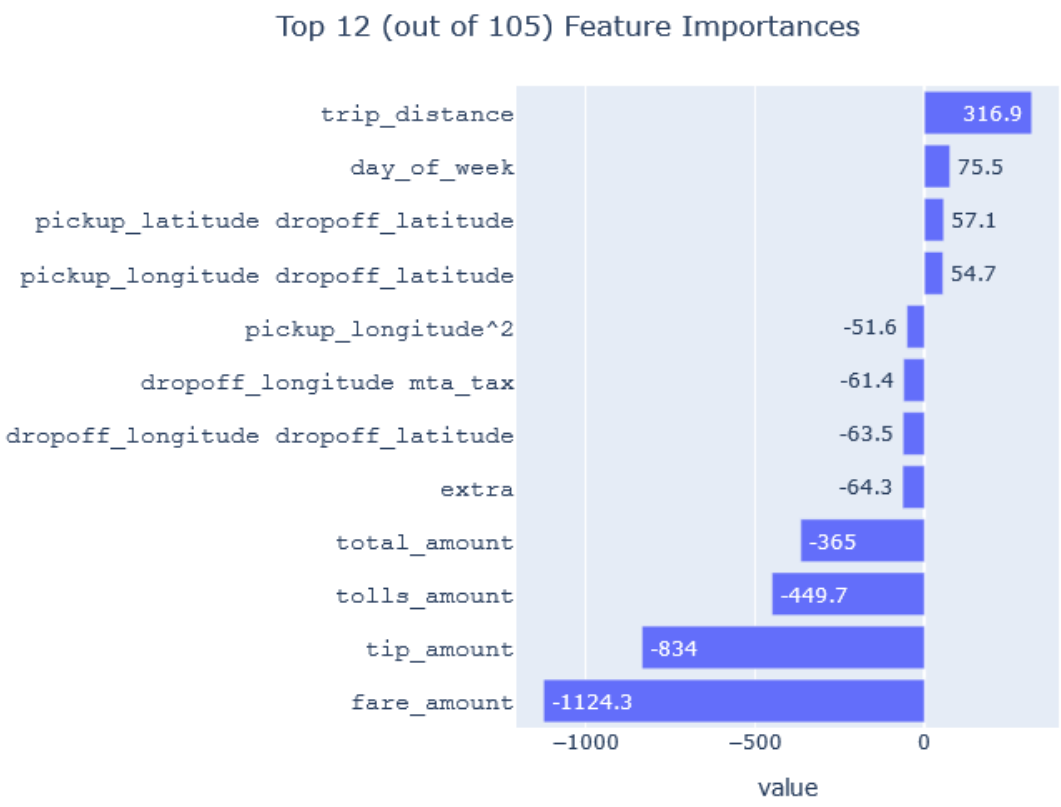
Key Findings and Insights

We explored the feature importance for each model and identified some of the features that each model used in making their predictions as shown below:

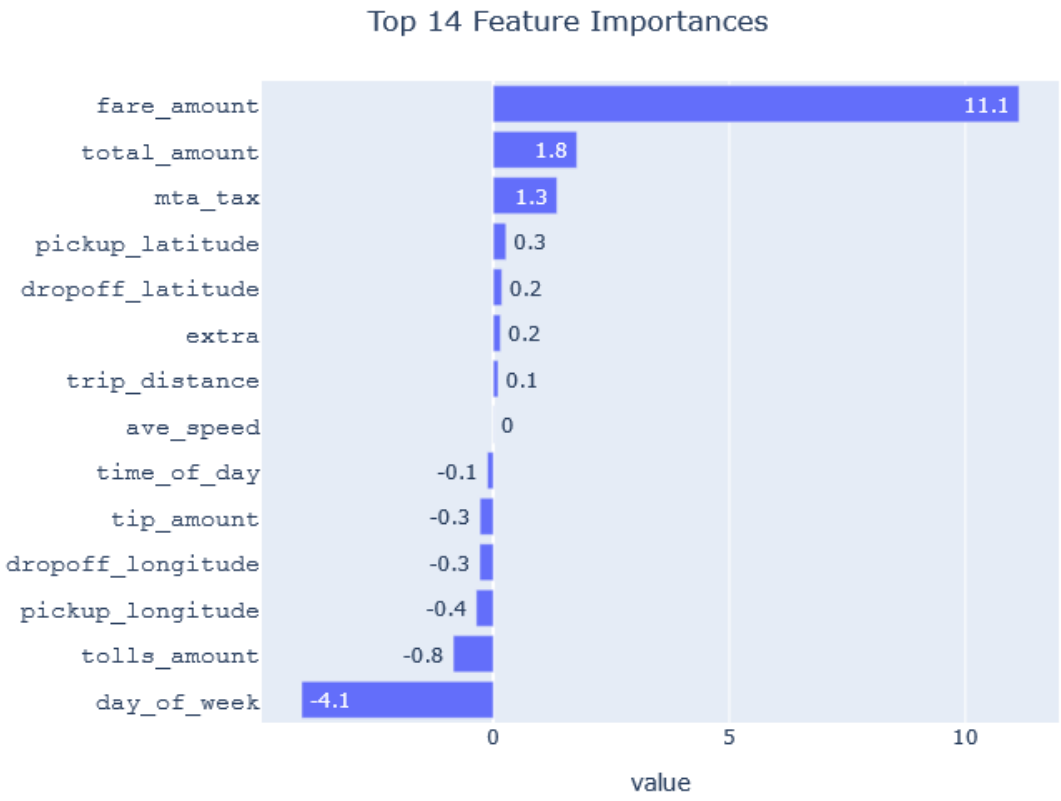
1. Simple Linear Regression model without polynomial features



2. Linear Regression model with polynomial features: This model has about 105 features and we chose only 12 with feature importance greater than 50.



3. Ridge model without polynomial features and used ***GridSearchCV*** to select the best alpha to avoid overfitting.



Model Selection

Based on the score and rmse error we propose to use the Linear Regression model with polynomial features.

Next Steps in model improvement

The Linear Regression model with polynomial features chosen above has a lot of features (105), we will explore other models that can perform as well with a minimum number of features.

Summary

This data set has very rich and important features used to build this machine learning model for predicting trip duration.